Journal of Computing and Information Technology - CIT 18, 2010, 3, 275–283
doi:10.2498/cit.1001737

Keystroke Saving in a Language with Highly Transparent Orthography

Francesco Curatelli[1] and Chiara Martinengo[2]

[1]Department of Biophysical and Electronic Engineering (DIBE), University of Genova, Italy
[2]Department of Mathematics (DIMA), University of Genova, Italy

This paper proposes a pseudo-syllabic soft keyboard for the Croatian language. The orthogonal keyboard layout makes it possible to improve typing efficiency in terms of keystroke saving, and is based on a highly ordered arrangement of pseudo-syllabic keys. The positions of the consonant and vowel graphemes that constitute a pseudo-syllable are used to access it orthogonally and independently of each other. This allows the user to input a pseudo-syllable with a lower cognitive load than with non-orthogonal 2-D layouts. Moreover, due to the almost perfect transparency of the language, a pseudo-syllable to be input can be accessed fast and with a reduced cognitive load starting from its phonetic sounds. The results of the present study show that the obtainable keystroke savings are comparable with those scored by word prediction tools with one suggestion, i.e., those requiring only a moderate cognitive load by the user.

Keywords: human-computer interaction, soft keyboard, transparent orthography, orthogonal keyboard, keystroke saving

1. Introduction

One of the major research fields in Human-Computer Interaction (HCI) concerns the study and implementation of new text entry methods, which allow users to input more *words per minute* (*WPM*) with hardware or on-screen soft keyboards [13].

Indeed, standard hardware keyboards and touch-screen soft keyboards (where typing is done with a stylus or a finger) allow *WPM* values exceeding 40 and 30, respectively [7]. However, only expert users can achieve these speeds, and anyway typing speeds greatly decrease when mice or pointing devices (such as tracking ball, graphics tablet or eye-tracking) are used [14]. Moreover, non-expert and motor-impaired users typically obtain significantly lower values; only rarely, Alternative and Augmentative Communication (AAC) users are able to input (letter-by-letter) more than 10-15 *WPM*, whereas in many cases, because of the motor impairment, typing speeds can be as low as 1 *WPM* or even less. [20].

Although the use of optimised hardware and soft keyboards increase typing speed [11, 12], only expert users fully exploit this possibility, being able to access keys without the need of visual searching the keys on the keyboard layout. Instead, all other users typically need a significant time to visually scan the keyboard to access a key. This time is suitably modelled by the Hick-Hyman law: $T_v = a' + b' \log_2(n)$, where n is the number of the keys, a' is the response time and $1/b'$ is the bandwidth [18].

So, for non-expert and impaired users the best solution is to improve input speed by reducing the number of keys the user has to select to input the text. This reduction can be measured as the mean number of key selections needed to input a character $KSPC = \#Keys/\#Chars$ [13]. This means that the user is able to input $\#Chars$ characters by selecting $\#Keys < \#Chars$ keys, so obtaining a high *keystroke saving rate* (or simply: *saving rate*). $KSR = 100 \times (\#Chars - \#Keys)/\#Chars$. Obviously, the standard letter-by-letter input methods are characterised by $KSPC = 1$ and $KSR = 0$.

To this aim, a common approach is to use text prediction software tools, which provide the user a window with one or more suggestions for completing the current or, sometimes, the next word [4, 16, 15, 22]. In some cases, the

suggestions are not provided within a window, but displayed as moving texts on the screen area, so that the user selects the desired texts by forcing them to pass through a target zone [23]. With prediction tools savings up to 60% can be reached, but only when 5 or more suggestions are displayed [21, 19]. Instead, savings up to 37.1% are obtained for English with only one suggestion [19], and similar values (35%) are obtained with inflected languages, such as Dutch, French, German and Swedish [15]. This limitation cannot be ignored because, in any predictive tool, the higher the number of the proposed suggestions, the higher is the cognitive load required by the users, who have to visually scan the suggestion list to find the right word (if any). This is the reason why the high *KSR* values obtainable with prediction tools do not yield comparably high communication rates [20]. In fact, although the adoption of good word prediction systems can be often advantageous, this is not always the case. For example, some studies on the use of query suggestions in mobile phones have shown that using suggestions nearly halved the number of pressed key to enter a query, but the corresponding measured time was not reduced at all because of the cognitive load required by the user to read and evaluate the list of suggestions [10].

Another way to increase *KSR* is to implement on-screen keyboards where the keys do not hold single letters, but syllables or sub-syllables of the target language. This solution can be highly advantageous because a clever choice of the syllabic entities that are present in the keyboard makes it possible to score high *KSR* values, which is the main target of any new text entry paradigm. On the other hand, as just shown for word prediction, keystroke saving is fully exploited if only a moderate cognitive load is required by the user for searching and assembling the sequence of keys in the keyboard layout.

This has been the rationale for the introduction of a novel *orthogonal keyboard* paradigm, which is based on the definition of *pseudo-syllables* (*p-syllables*) as basic text entry units [2]. A target p-syllable is accessed by composing, orthogonally and independently of each other, the positions of its consonant and vowel graphemes (column and row, respectively), without the need to visually scan the whole keyboard or to memorise it. According to the Hick-Hyman law, this implies a reduction of the number of the items that have to be inspected or memorised and, consequently, of the required cognitive load and search time.

This obviously holds for a whichever alphabetical language, but especially for languages with transparent orthographies, i.e., with highly regular correspondence between orthography and phonetics. In fact, the presence of an almost one-to-one correspondence between phonemes and graphemes, which is typical of transparent languages, exempts the user from the additional cognitive load that is required to perform the complex sound to text translation needed in non-transparent languages such as English. Moreover, the presence of a direct and intuitive phoneme to grapheme association makes it possible for an efficient use of the orthogonal keyboard, not only by normal users (expert and non-expert), but also, and mainly, by motor-impaired users and by users who are affected by difficulty in reading and writing (i.e., dyslexia and dysgraphia).

The orthogonal keyboard paradigm has been suitably applied to Italian and Spanish [2], which are two languages with transparent orthographies, and, more recently, also to a highly non-transparent language such as English [3]. In this last case, similar keystroke savings have been obtained with a clever choice of the p-syllable set, which makes it possible to directly input many monosyllabic English words [3].

However, since Italian and Spanish are far from being really *perfect* transparent languages, from the beginning of our research in this field we have been intrigued by the possibility to apply the orthogonal paradigm to languages which do as much as possible approximate perfect transparency. In particular, we were fascinated by the possible application to Slavic languages such as Croatian and Serbian, which have two almost perfect transparent orthographies, derived from a clever choice of the two alphabets used, i.e., the modified Gaj's Latin alphabet [5] for Croatian and the Srpska Ćirilica for Serbian. Moreover, as the Croatian alphabet is also one of the two scripts used for the Bosnian and Serbian languages, and a reduced version is used for the Slovenian language, Croatian has been chosen as target language for the study which is described in the present paper.

To this aim, the characteristics of Croatian have been analysed and an orthogonal keyboard layout has been designed according to the frequency statistics of the language. The suitability of the proposed layout has been then tested with results that are comparable with those obtained with Italian and Spanish keyboards.

In Section 2, the main characteristics of the orthogonal keyboard paradigm are outlined, while Section 3 outlines the statistical analysis of the language and the design of the Croatian orthogonal keyboard layout. Section 4 outlines the experimental results and conclusions.

2. The Orthogonal Pseudo-syllabic Model

The orthogonal text entry paradigm is based on the definition of a keyboard layout which makes it possible to input *dominant* and *subordinate p-syllables*. The dominant p-syllables are directly selected without using any shift key, while the subordinate ones need the selection of specific shift keys. Both types are accessible without the need to perform a complete visual exploration of the orthogonal keyboard, because the text units are assigned to the keyboard keys so that their horizontal and vertical positions are easily inferrable from the graphemic structure of the related p-syllabic entity. In fact, each column of the keyboard is assigned to a specific *consonant grapheme* $\hat{C}$, which can be either a single consonant or a sequence of consonants denoting (in a transparent language) a single phoneme. Instead, each row is assigned to a specific *vowel grapheme* $\hat{V}$, which can be either a single vowel or semi-vowel or a sequence of them. In this way, the p-syllable that contains both the vowel grapheme of row x and the consonant grapheme of column y can be accessed by the direct composition of the consonant and vowel positions, which are independent of each other (orthogonal). Anyway, only one action is needed to input dominant or subordinate p-syllables (provided that the related shift key is concurrently selected). All other syllables and pseudo-syllables can be selected with a sequence of dominant or subordinate p-syllables, so requiring two or (sometimes) more actions.

Given a language L, we start by defining the *complete consonant* and *vowel sets* $\widehat{CS_L}$ and $\widehat{VW_L}$, which contain the most used consonant and vowel graphemes $\widehat{C}_i$ and $\widehat{V}_i$, respectively ($n_{\widehat{CS}_L} = |\widehat{CS}_L|$ and $n_{\widehat{VW}_L} = |\widehat{VW}_L|$ are the set cardinalities). The two sets also contain the empty grapheme ϵ, so that the x-y grapheme composition can build purely vowel and consonant p-syllables too. So, the *complete p-syllable set* PS_L (with cardinality $n_{PS_L} = |PS_L|$) contains any possible string $\widehat{C}_i\widehat{V}_i$ and $\widehat{V}_i\widehat{C}_i$. The second ones are subordinate p-syllables and are selectable by means of the *forward/reverse shift* (*FR-Sh*).

As already stated, this scheme allows users to access the target p-syllable without the need of visually scanning the whole keyboard or memorizing it. This implies a reduction of the required cognitive load and search time, due to the smaller number of the items to be considered. Therefore, also non-expert users can obtain better access times than with non-ordered p-syllable layouts. It is worth noting that this approach is completely different from the typical approach found in literature, which is based on the selection of several single-letter keys to build syllabic entities [9].

In many cases, the direct use of the complete p-syllable set would yield a too large keyboard layout; when this is the case, the size of the orthogonal array must be reduced by applying one or more optimisation steps to generate the final orthogonal keyboard layout. First, it can be verified if it is convenient to eliminate columns or rows whose p-syllables have low marginal frequencies in the target language (*column deletion (CD)* and *row deletion (RD)*). Second, it can be verified if it is convenient to fold columns containing p-syllables with low marginal frequencies, by assigning them to other columns (*column folding (CF)*), and merge rows containing p-syllables with low marginal frequencies, by assigning them to other rows (*row folding (RF)*). The selection of a subordinate key belonging to one of the folded entities is done by selecting one of two specific shifts (*CF-Sh* for *CF* optimisation or *RF-Sh* for RF optimisation), which are unique for all the folded columns and folded rows. Another possible optimisation concerns merging together clusters of two or more columns if each column contains a subset of rows with rarely used p-syllables, and the subset belongs to rows different from those of

all the other columns of the same cluster (*Column merging (CM)*).

When CD and RD optimisations are applied, there is a reduction in the consonant and vowel graphemes, thus two optimised sets for consonant and vowel graphemes, $\widehat{CS}_L^o$ and $\widehat{VW}_L^o$, and, consequently, an *optimised p-syllable set* PS_L^o are obtained.

Is it worth noting that in defining the orthogonal keyboard layout only the optimisation steps should be applied which are strictly needed to obtain a suitable keyboard size for the target language. As it will be shown in the next Section, Croatian orthography does not require an extensive application of the optimisation steps. In fact, the definition of the orthogonal keyboard will require only the CF optimisation.

3. The Croatian Orthogonal Keyboard

As already stated, we have chosen Croatian because of its almost perfect transparent orthography. In fact, the native Croatian alphabet is composed of 30 graphemes, with a one-to-one correspondence with the phonemes of the language: *a, b, c, č, ć, d, dž, đ, e, f, g, h, i, j, k, l, lj, m, n, nj, o, p, r, s, š, t, u, v, z, ž*; i.e.:

a) 22 one-letter graphemes of the English alphabet (in fact, *q, w, x* and *y* do not belong to the Croatian alphabet);

b) 5 specific one-letter consonants with diacritics (*č, ć, đ, š* and *ž*);

c) 3 specific two-letter consonants (digraphs) which denote specific sounds and are treated in all effects as separate letters (*dž, lj*, and *nj*) (they correspond exactly to one-letter graphemes of the Serbian Cyrillic alphabet).

This alphabet is also called Gaj's Latin alphabet, after the Croatian linguist Ljudevit Gaj who in 1830 set the Croatian alphabet starting from the Latin alphabet [5]; the only difference from the actual script consisted in the use of a further digraph *dj* in place of the actual letter *đ*. It is worth noting that the Croatian alphabet is also one of the two scripts used for the Bosnian and Serbian languages; moreover, a reduced version is used for the Slovenian language.

Concerning the missing Latin letters (*q, w, x* and *y*), although they are used only to write foreign words, their increasing presence in modern texts (such as newspaper articles, textbooks, and so on) has suggested not to discard them from the orthogonal layout. Instead, a notable characteristic of the Croatian language is that doubled consonants are very rarely used; for this reason, they have not been included in the orthogonal layout. Finally, the double grapheme *qu* has been used in place of *q* because in almost all the words (both Croatian and foreign) the simple consonant *q* is followed by *u*. Therefore, the starting (complete) consonant and vowel sets have been defined as follows:

$$\widehat{CS}_{\text{hr}} = \{\ \epsilon,\ b,\ c,\ č,\ ć,\ d,\ dž,\ đ,\ f,\ g,\ h,\ j,\ k,\ l,\ lj,\ m,\ n,\ nj,\ p,\ qu,\ r,\ s,\ š,\ t,\ v,\ w,\ x,\ y,\ z,\ ž\ \}$$

$$\widehat{VW}_{\text{hr}} = \{\epsilon,\ a,\ e,\ i,\ o,\ u\ \}$$

Because $|\widehat{CS}_{\text{hr}}| = 30$ and $|\widehat{VW}_{\text{hr}}| = 6$, the resulting keyboard would be too wide for an efficient use. The reduction of the keyboard width has been done by identifying a suitable set of less frequently used p-syllables and then applying a folding optimisation step which sets them as subordinate p-syllables of the orthogonal layout. To this aim, we have generated a detailed statistic of the p-syllable frequencies starting from the data present in the *Croatian Language Repository - newspapers sub-corpus* [8], which gives the relative frequencies of 710,177 different words in texts taken from Croatian newspapers [1]. In particular, we have taken into account the first 120,000 most frequent words, whose relative frequencies constitute more than 97% of the total word occurrences.

Because the dominant p-syllables have the structures $\hat{C}$, $\hat{V}$, or $\hat{C}\hat{V}$, we have extracted the frequency statistics concerning the 34 single graphemes $\hat{G} = \hat{C}$ or $\hat{V}$, and the 145 double graphemes $\hat{C}\hat{V}$, by computing, for each p-syllable, the sum of the weighted relative frequencies of the words in which the p-syllable is present. In other words, the relative frequency of each word has been taken n times, where n is the number of occurrences of the p-syllable in that word; in this way, a more realistic statistics of real frequencies of the graphemes has been obtained. The statistics concerning the single and double grapheme p-syllables are outlined in Table 1 and Table 2, respectively.

$\hat{G}$	$\%\hat{G}$	$\hat{G}$	$\%\hat{G}$	$\hat{G}$	$\%\hat{G}$	$\hat{G}$	$\%\hat{G}$
a	59.64	j	19.65	b	7.89	f	1.16
i	50.84	k	19.13	č	4.91	đ	1.04
o	46.90	d	18.30	c	4.90	y	0.14
e	43.90	v	18.04	š	4.49	w	0.13
n	29.69	m	15.68	h	4.08	dž	0.04
r	26.98	p	15.26	ć	3.92	x	0.04
t	24.57	l	13.99	nj	3.79	q	0.01
s	24.51	z	9.46	lj	2.75		
u	22.65	g	8.61	ž	2.74		

Table 1. Single grapheme % statistics.

The CF optimisation defines the final orthogonal layouts for the dominant and the subordinate p-syllables that require the selection shift *CF-Sh* be *on*. As the dominant p-syllables are selectable directly in the layout, they have been chosen according to the frequencies outlined in Table 1 and Table 2, which show very low frequencies for p-syllables involving the consonants *y, w, dž, x* and *q*. In fact, the relative frequencies in words of these consonants are strictly less than 0.15%, whereas all other consonants have values greater than 1%. This consideration has suggested to select as consonants for the dominant p-syllables the 24 consonants with frequencies exceeding 1%, i.e., all but the 5 consonants above. In this way, the size of the orthogonal keyboard is $(n_x \times n_y) = (25 \times 6)$, and the 25 columns of the orthogonal keyboard are assigned to the dominant consonants, lexicographically ordered one after the other (the empty grapheme ϵ is put at the beginning, so corresponding to the column of the vowels): ϵ, *b, c, č, ć, d, đ, f, g, h, j, k, l, lj, m, n, nj, p, r, s, š, t, v, z, ž.*

All other p-syllables, containing the consonants *y, w, dž, x* and *qu*, are set as subordinate p-syllables (whose selection requires that *CF-Sh* = *on*) and are allocated to 5 columns.

To render the cognitive task of the user as easy as possible, these 5 consonants have been lexicographically ordered one after the other; moreover *dž* and *qu* have been allocated to columns whose dominant consonants are lexicographically close to their original position in the unfolded configuration:

$\hat{C}$	$\%\,\hat{C}a$	$\%\,\hat{C}e$	$\%\,\hat{C}i$	$\%\,\hat{C}o$	$\%\,\hat{C}u$	$\hat{C}$	$\%\,\hat{C}a$	$\%\,\hat{C}e$	$\%\,\hat{C}i$	$\%\,\hat{C}o$	$\%\,\hat{C}u$
b	1.13	0.63	2.31	1.21	0.45	n	7.95	3.92	6.65	4.96	1.18
c	0.88	0.73	2.33	0.08	0.21	nj	0.99	1.62	0.58	0.08	0.40
č	0.75	1.07	1.10	0.04	0.24	p	1.30	0.51	0.78	5.05	0.67
ć	0.57	1.34	0.71	0.03	0.18	q	0.00	0.00	0.00	0.00	0.00
d	4.28	1.15	2.62	2.34	0.89	r	7.24	4.82	3.97	3.09	1.39
dž	0.00	0.01	0.01	0.00	0.00	s	1.82	2.40	1.06	0.43	2.20
đ	0.25	0.37	0.05	0.00	0.33	š	0.43	0.70	0.44	0.03	0.06
f	0.10	0.18	0.35	0.20	0.04	t	4.40	2.48	5.32	3.34	1.31
g	1.29	0.35	0.38	2.15	0.59	v	4.33	2.37	3.26	3.04	0.42
h	0.23	0.13	0.18	0.35	0.07	w	0.03	0.01	0.01	0.00	0.00
j	2.36	11.58	1.38	0.46	1.5	x	0.00	0.01	0.00	0.00	0.00
k	4.06	1.31	1.63	6.37	1.53	y	0.01	0.00	0.00	0.02	0.00
l	3.73	1.29	4.38	2.21	0.84	z	3.73	0.52	0.84	0.32	0.27
lj	0.64	0.90	0.34	0.04	0.45	ž	0.55	0.78	0.60	0.00	0.20
m	3.68	1.81	1.73	2.12	0.44						

Table 2. $\hat{C}\hat{V}$ % statistics.

dž => col 6 (*d*), *qu* => col 18 (*p*),

w => col 23 (*v*), *x* => col 24 (*z*),

y => col 25 (*ž*).

This CF mapping yields a subordinate layout in which only 5 columns are assigned; this means that 19 columns are potentially available for the allocation of $19 \times 6 = 114$ additional strings (*added words*). As our aim was to explore the potentiality of the orthogonal paradigm, this additional improvement has not been addressed in the present work. However, further gains in terms of keystroke saving could be easily obtained by allocating, as added words, a suitable set of syllables and words chosen among the most frequent words of the language, by using the data provided, for example, by the Croatian Language Repository [1].

The final keyboard configurations are outlined in Tables 3 to 5. Table 3 and 4 outline the layouts of the Croatian orthogonal keyboard for the dominant p-syllables and the reverse subordinate p-syllables with *FR-Sh* = *on*, while Table 5 outlines the layout for the subordinate p-syllables with *CF-Sh* = *on*.

It is worth noting that the use of dominant and subordinate p-syllables with *FR-Sh* = *on* makes it possible to insert the graphemic sequences *VCCV* and *CVVC* with only two actions. This is very important because in the Croatian language, as well as in almost all Slavic languages, the first graphemic structure is very frequent.

	b	c	č	ć	d	đ	f	g	h	j	k	l	lj	m	n	nj	p	r	s	š	t	v	z	ž
a	ba	ca	ča	ća	da	đa	fa	ga	ha	ja	ka	la	lja	ma	na	nja	pa	ra	sa	ša	ta	va	za	ža
e	be	ce	če	će	de	đe	fe	ge	he	je	ke	le	lje	me	ne	nje	pe	re	se	še	te	ve	ze	že
i	bi	ci	či	ći	di	đi	fi	gi	hi	ji	ki	li	lji	mi	ni	nji	pi	ri	si	ši	ti	vi	zi	ži
o	bo	co	čo	ćo	do	đo	fo	go	ho	jo	ko	lo	ljo	mo	no	njo	po	ro	so	šo	to	vo	zo	žo
u	bu	cu	ču	ću	du	đu	fu	gu	hu	ju	ku	lu	lju	mu	nu	nju	pu	ru	su	šu	tu	vu	zu	žu

Table 3. Croatian orthogonal keyboard for dominant p-syllables.

	b	c	č	ć	d	đ	f	g	h	j	k	l	lj	m	n	nj	p	r	s	š	t	v	z	ž
a	ab	ac	ač	ać	ad	ađ	af	ag	ah	aj	ak	al	alj	am	an	anj	ap	ar	as	aš	at	av	az	až
e	eb	ec	eč	eć	ed	eđ	ef	eg	eh	ej	ek	el	elj	em	en	enj	ep	er	es	eš	et	ev	ez	ež
i	ib	ic	ič	ić	id	iđ	if	ig	ih	ij	ik	il	ilj	im	in	inj	ip	ir	is	iš	it	iv	iz	iž
o	ib	ic	ič	ić	id	ođ	if	ig	ih	ij	ik	il	ilj	im	in	inj	ip	ir	is	iš	it	iv	iz	iž
u	ub	uc	uč	uć	ud	uđ	uf	ug	uh	uj	uk	ul	ulj	um	un	unj	up	ur	us	uš	ut	uv	uz	už

Table 4. Croatian orthogonal keyboard for subordinate p-syllables (*FR-Sh* = *on*).

	b	c	č	ć	dž	đ	f	g	h	j	k	l	lj	m	n	nj	qu	r	s	š	t	w	x	y
a	ba	ca	ča	ća	dža	đa	fa	ga	ha	ja	ka	la	lja	ma	na	nja	qua	ra	sa	ša	ta	wa	xa	ya
e	be	ce	če	će	dže	đe	fe	ge	he	je	ke	le	lje	me	ne	nje	que	re	se	še	te	we	xe	ye
i	bi	ci	či	ći	dži	đi	fi	gi	hi	ji	ki	li	lji	mi	ni	nji	qui	ri	si	ši	ti	wi	xi	yi
o	bo	co	čo	ćo	džo	đo	fo	go	ho	jo	ko	lo	ljo	mo	no	njo	quo	ro	so	šo	to	wo	xo	yo
u	bu	cu	ču	ću	džu	đu	fu	gu	hu	ju	ku	lu	lju	mu	nu	nju	quu	ru	su	šu	tu	wu	xu	yu

Table 5. Croatian orthogonal keyboard for subordinate p-syllables (*CF-Sh* = *on*).

4. Experimental Results and Conclusions

The Croatian orthogonal keyboard has been evaluated by using the following parameters. $N_W, N_\alpha, N_O, N_{Sp}$ are the numbers of words, alphanumeric, non-alphanumeric, and space characters in the text; $N_C = N_\alpha + N_O$ is the total number of characters in the text; K_α is the number of alphanumeric keys typed on the keyboard; S_{Sh} and S_K are the numbers of capital letters and *FR-Sh* shifts.

The typing gains are measured by 3 types of *KPSC* (*keystrokes per character*) and *KSR* (*keystroke saving rate*) parameters. $KSPC_\alpha = K_\alpha/N_\alpha$ is the mean number of the alphanumeric keys to input an alphanumeric character, and $KSR_\alpha = 100 \times (1 - KSPC_\alpha)$ is the percentage of the alphanumeric keys saved by using the orthogonal keyboard.

$KSPC_\beta = (K_\alpha + N_O - N_{Sp})/(N_C - N_{Sp})$ is the mean number of the non-space keys to input a non-space character, and $KSR_\beta = 100 \times (1 - KSPC_\beta)$ is the percentage of the non-space keys saved by using the orthogonal keyboard.

Finally, $KSPC_\gamma = (K_\alpha + N_O)/N_C$ is the mean number of keys to input a whichever character, and $KSR_\gamma = 100 \times (1 - KSPC_\gamma)$ is the percentage of the keys (of any type) saved by using the orthogonal keyboard.

To test the keyboard layout, some Croatian texts have been automatically analysed to find how much improvement can be obtained with respect to the classical, letter-by-letter approach. The texts were downloaded from the electronic versions of the popular Croatian magazine *Globus* [6].

The obtained results (outlined in Table 6), show significant keystroke savings (greater than 40.5%) for alphanumeric characters, and only slightly lower values (greater than 38.8%) for non-spaces characters. This means that the additional use of clever algorithms for the automatic addition of spaces could easily improve the already good (greater than 32.2%) keystroke savings obtained for all characters. These results are impressive, especially if they are compared with the only slightly better keystroke savings obtained with Italian and Spanish [2]. In fact, these two Neolatin languages are structurally characterised by very high frequencies of *CV* sequences, whereas in Croatian (and in almost all Slavic languages) triple and quadriple consonants sequences occur very frequently. Probably, these high gains can be ascribed to the fact that in Croatian most of the more frequent words are monosyllabic words with *CV* structure. In fact, according to the used Croatian Corpus, the ten most frequent words are, in order: *je, u, i, se, da, na, za, su, a, od* [1].

The obtained keystroke savings are comparable with those obtained, for English and highly inflected languages with one-suggestion word prediction tools [19, 15], which, as stated in Section 1, require only a moderate cognitive load. This is also the case with the orthogonal keyboard, which is a text entry method that requires a lower cognitive load than non-orthogonal 2-D

Text	N_W	N_C	N_α	N_O	N_{Sp}	S_{Sh}	S_K	K_α
globus-1	1341	8357	6724	1633	1328	152	278	3995
globus-2	1223	7098	1514	5584	1197	169	134	3258
globus-3	1181	6701	5237	1464	1163	158	159	3077

(a)

Text	$KSPC_\alpha$	KSR_α	$KSPC_\beta$	KSR_β	$KSPC_\gamma$	KSR_γ
globus-1	0.5941	40.58	0.6117	38.82	0.6734	32.65
globus-2	0.5834	41.65	0.6058	39.41	0.6723	32.76
globus-3	0.5875	41.24	0.6099	39.00	0.6776	32.23

(b)

Table 6. KSR Statistics for the Croatian orthogonal keyboard.

layouts. As seen, this is due to the fact that the visual search task and, consequently, the cognitive load, can be modelled by using the well established Hick-Hyman law.

Besides the reduction of the cognitive load due to the characteristics of the orthogonal scheme, the question may be raised if the same can be said for an additional reduction due to the transparency level of the target language. Certainly, in a perfect transparent language such as Croatian, as well as almost always in languages such as Italian and Spanish, there is a one-to-one correspondence between phonemes and graphemes. For this reason, the translation task is very simple, and the related cognitive load is low, whereas in English the same task is much more complex, thus requiring a higher cognitive load. But, one may wonder if this reduction could be accurately measured as a function of language transparency, i.e., by means of some direct or indirect objective measure [17]. At first sight, a possible indirect and objective measure could be the typing speed, since there is an intuitive correlation between *WPM* and cognitive load. However, there are also other important factors, such as differences in orthography, grammatical rules and inflection of the languages involved, which do not allow us to tell if this correlation can be accurately modelled or not. Thus, the answer to this interesting issue would require specific and detailed experiments, which were not among the aims of the present work.

In conclusion, the present study has shown that the orthogonal paradigm is well suited to obtain high keystroke savings for a highly transparent Slavic language such as Croatian. This is a promising result, also because we can argue that this result can be automatically extended to Serbian, due to its almost perfect alphabetic correspondence and language similarity with Croatian. Thinking along the same lines, it would be useful to explore the application of the orthogonal paradigm to Cyrillic alphabets, because of their use in Serbian, Bulgarian, Macedonian, Russian and many other Eastern languages. Another interesting target language is Polish, which is not only a highly transparent language, but is also the most widespread Slavic language written with the Latin alphabet.

References

[1] The Croatian language repository. Inst. of Croatian Language and Linguistics, 2006 (downloaded: August 2009). `http://riznica.ihjj.hr`

[2] F. CURATELLI, C. MARTINENGO, A powerful pseudo-syllabic text entry paradigm. *Int. Journal of Human-Computer Studies*, 64 (5), 475–488, 2006.

[3] F. CURATELLI, C. MARTINENGO, English orthogonal keyboard for efficient text entry. Submitted for publication, 2010.

[4] J. J. DARRAGH, I. H. WITTEN, Adaptive predictive text generation and the reactive keyboard. *Interacting with Computers*, 3 (1), 27–50, 1991.

[5] L. GAJ, *Kratka osnova horvatsko-slavenskog pravopisanja* (Brief basics of the Croatian-Slavonic orthography), Buda, 1830.

[6] Globus magazine, electronic edition, 2009 (downloaded: October 2009). `http://www.globus.com.hr/`

[7] K. GO, Y. ENDO, A touchscreen software keyboard for finger typing. In *Advances in Human-Computer Interaction* (A. Lazinica Ed.), 2008, pp. 287–296.

[8] I. P. JAZBEC, T. STOJANOV, The Croatian Language Corpus. Presented at the *Proceedings of the CECL 2006 Conf.*, 2006, Zadar, Croatia.

[9] A. JOSHI ET AL., Keylekh: a keyboard for text entry in indic scripts. Presented at the *Proceedings of the CHI 04 Conf.*, 2004, Vienna, Austria, pp. 928–942.

[10] M. KAMVAR, S. BALUJA, Query suggestions for mobile search: understanding usage patterns. Presented at the *Proceedings of the CHI 08 Conf.*, 2008, Florence, Italy, pp. 1013–1016.

[11] G. W. LESHER, B. J. MOULTON, D. J. HIGGINBOTHAM, Optimal character arrangements for ambiguous keyboards. *IEEE Transactions on Rehabilitation Engineering*, 6 (4), 415–423, 1998.

[12] I. S. MACKENZIE, S. X. ZHANG, The design and evaluation of a high-performance soft keyboard. Presented at the *Proceedings of the CHI 99 Conf.*, 1999, Pittsburgh, USA, pp. 25–31.

[13] I. S. MACKENZIE, R. W. SOUKOREFF, Text entry for mobile computing: models and methods, theory and practice. *Human-Computer Interaction*, 17 (2–3), 147–198, 2002.

[14] P. MAJARANTA, K. RÄIHÄ, Text entry by gaze: utilizing eye-tracking. In *Text entry systems: Mobility, accessibility, universality* (I.S. MacKenzie and K. Tanaka-Ishii Eds.), 2007, pp. 175–187.

[15] J. MATIASEK, M. BARONI, H. TROST FASTY, A multilingual approach to text prediction. Presented at the *Proceedings of the ICCHP 02 Conf.*, 2002, Linz, Austria, pp. 243–250.

[16] R. ROSENFELD, A maximum entropy approach to adaptive statistical language modelling. *Computer, Speech and Language*, 10, 187–228, 1996.

[17] P. SCHMUTZ, S. HEINZ, Y. METRAILLER, K. OPWIS, Cognitive load in ecommerce applications-measurement and effects on user satisfaction. *Advances in Human-Computer Interaction*, ID 121494, 9 pages, 2009.

[18] R. W. SOUKOREFF, Text entry for mobile systems: models, measures, and analyses for text entry research. M.Sc. Thesis York University, Canada, 2002.

[19] K. TRNKA, K. F. MCCOY, Evaluating word prediction: framing keystroke savings. Presented at the *Proceedings of the ACL 08 Conf.*, 2008, Columbus, USA, pp. 261–264,

[20] K. TRNKA, J. MCCAW, D. YARRINGTON, K. F. MCCOY, User interaction with word prediction: the effects of prediction quality. *ACM Trans. on Accessible Computing*, 1 (3), 17:1–34, 2009.

[21] P. A. VÄYRYNEN, K. NOPONENA, T. SEPPÄNEN, Analysing performance in a word prediction system with multiple prediction methods. *Computer Speech & Language*, 21 (3), 479–491, 2007.

[22] T. WANDMACHER, J. Y. ANTOINE, F. POIRIER SIBYLLE, A system for alternative communication adapting to the context and its user. Presented at the *Proceedings of the ASSETS 07 Conf.*, 2007, Tempe, USA, pp. 203–210

[23] D. J. WARD, A. F. BLACKWELL, D. J. MACKAY DASHER, A gesture-driven data entry interface for mobile computing. *Human-Computer Interaction*, 17 (2–3), 199–228, 2002.

Received: October, 2009
Revised: February, 2010
Accepted: March, 2010

Contact addresses:

Francesco Curatelli
DIBE–University of Genova
Via Opera Pia 11/A
16145 Genova, Italy
e-mail: curatelli@unige.it

Chiara Martinengo
DIMA–University of Genova
Via Dodecaneso 35
16146 Genova, Italy

FRANCESCO CURATELLI is a professor of electronics at the University of Genova – Department of Biophysical and Electronic Engineering (DIBE), Genova, Italy. His research activities include the study and development of CAD tools, techniques and algorithms for hardware/software real-time systems, educational methodologies for teaching mathematics to impaired children, human-computer interaction and assistive technology.

CHIARA MARTINENGO is a researcher in mathematics at the University of Genova – Department of Mathematics (DIMA), Genova, Italy. Her research activities include the study of several arguments in commutative algebra and algebraic geometry, educational methodologies for teaching mathematics to impaired children, human-computer interaction and assistive technology.